Chapter 9

A Multi-Agent Approach for Designing Next Generation of Air Traffic Systems

José Miguel Canino, Juan Besada Portas, José Manuel Molina and Jesús García

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/50284

1. Introduction

Current implementation of new technologies for Communication, Navigation, Surveillance and Air Traffic Management (CNS/ATM systems) along with computational improvements on airborne and ground systems developed in the last two decades, point out the need for more strategic navigation and air-traffic control procedures based on four-dimensional (position plus time) trajectories. Moreover, the CNS/ATM infrastructure will help to achieve more shared real-time information among aircraft, airlines and air-traffic services providers (i.e. Air Traffic Control –ATC- providers, meteorological information providers and air space resources information providers). Then, general requirements for a next-generation of Air Traffic Management (ATM) system is that entities must share real-time information about aircraft state and intentions, air-space state and resources, meteorological conditions and forecast, etc. From this information coordinated and strategic actions should be carried out by them in order to achieve efficient and free of conflict 4D trajectories.

Several systems have been used on last years to help controllers and pilots to take decisions in a more strategic way. Some examples of these systems are OASIS [1], MAESTRO [2], COMPAS [3], CTAS, [4], etc. In addition, in the PHARE program [5], on-board trajectory predictions were used to inform the air traffic controllers about aircraft intentions. Results of the previous proposal showed potential advantages of a more strategic navigation and Air Traffic Control (ATC) based on 4D trajectories.

However new efforts are still necessary for achieving an ATM system characterized by a greater aircraft autonomy to select an optimal flight path and by a higher automation of air-ground coordination tasks. Thus, the Free Flight operational concept [6] suggests a future ATM where the aircraft are only restricted by global goals that must ensure safe and efficient air traffic flows. By other hand, the Trajectory Based Operations (TBO) [7] attempts to give more specific to the free flight proposal. In this case TBO concept proposes that

coordination activities between ATM system elements are intended to achieve efficient and free of conflict 4D trajectories. In a hypothetical TBO scenario, aircraft will calculate their User Preference Trajectories (UPTs) taking into account air-space constraints. In addition, an ATC with the role of a central agent should drive air ground negotiation processes in order to provide free of conflict trajectories. Furthermore, under certain circumstances aircraft self-separation could be possible and air-air negotiation processes could be only supervised by the ATC. After a negotiation is performed the aircraft must fly the negotiated trajectories until a new set of trajectories were negotiated. Meanwhile, if emergence or contingence arises, a new negotiation process could be triggered.

In summary the new operational concepts propose:

- Four dimensional trajectory based operations defined by the aircraft position and time. These trajectories must suit the preferences of the aircraft while preserving the efficiency and safety of surrounding air traffic flow.
- Accessibility and distribution of updated data among all entities involved in flight operations.
- A more distributed reallocation of the roles of aircraft and services of air traffic control to achieve their respective goals, in contrast with the current scheme of responsibilities characterized by a ground-centralized monitoring and air traffic separation activities.

Application of these operational concepts by 2020+ is the key target of current research initiatives such as SESAR (Single European Sky ATM Research) and Next-Gen (Next Generation Air Transportation System) [8-9]. These research programs show that the set of activities aimed to validate procedures and technologies in order to implement the cited paradigm is diverse and extensive. To identify the interdependence of such activities, we propose a framework that classifies them according to a sequential time process (see Fig. 1). The first two groups of activities are referred to the analysis of the potential of CNS/ATM systems and as well as feasible operational concepts: i.e. Free Flight, TBO, etc. Proposals of operational concepts consist of generic specifications that requires of concrete procedures for conducting operations navigation and air-traffic control. Parallel to procedures design, on-board and ground systems and underlying mathematical models have to be also developed. Then, initial design of procedures and their associated systems can be considered as an iterative process that needs to be validated by means of analytical simulation. This process is a key issue as a previous stage to the Human-In-The-Loop (HITL) simulations and flight tests. HITL simulations and flight trials allows defining specific standards (e.g Procedures for Air Navigation Services –PANS- or for Required Performance Navigation –RNP-) for the actual implementation of the operational concept.

Several attempts for developing simulation and design analysis tools have been proposed [10-16]. Results of these studies show that it will be necessary more detailed and structured conceptual models to give support to analytical simulation tools to address the paradigm shift in the ATM procedures. Moreover, the close interdependence between coordination procedures, systems to execute them and their underlying mathematical models must be clearly set out in the conceptual model.

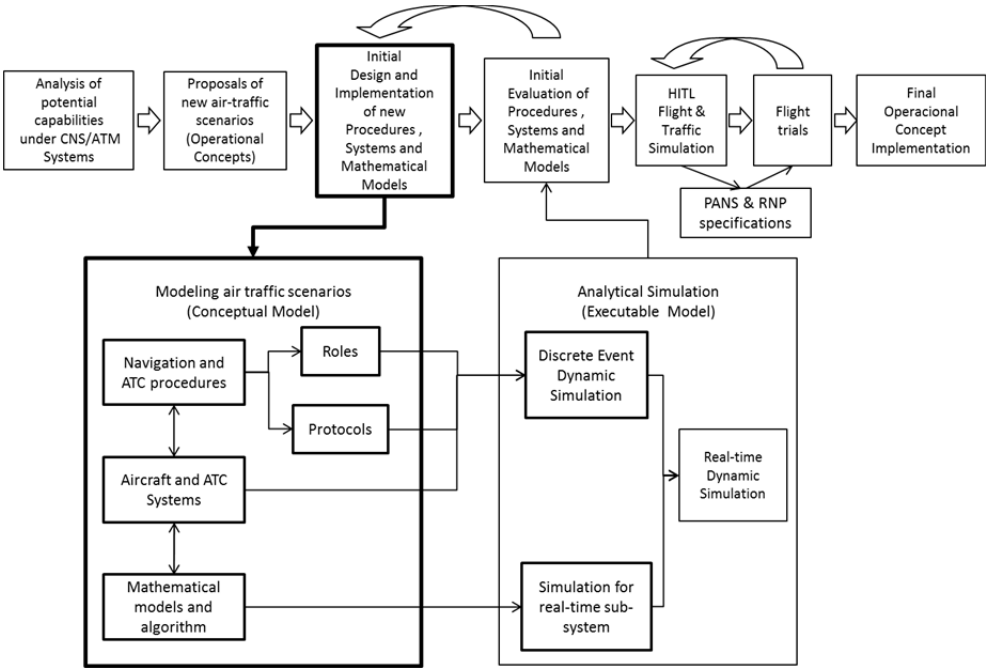

Figure 1. Research and development activities for new operational concepts

Fortunately, current state of the art of Multi-agent technologies provides methodological approaches and tools to develop such models. Then, the main goal of this chapter consists in illustrating how recent proposals in agent-oriented methodologies are a suitable approach for analysing and modelling new air traffic scenarios in order to obtain such conceptual models.

The chapter begins summarizing main contributions on modelling and simulations of future ATM. This review shows that new simulation platforms should be based on more detailed and structured conceptual models. In additions, it suggests benefices of multi-agent methodologies for approaching this problem. In section 3, multi-agent approaches for modelling several aspects of cited ATM are analysed. After a brief description of particularities of the multi-agent theory, a review of its most recent applications within the air traffic scope is presented. This exploration highlights the need of taking advantage of modern multi-agent technologies for developing robust and modular conceptual models of the new ATM. Section 4 presents the multi-agent methodologies as useful tools for analysing and modelling this ATM in order to facilitate the design and take key decisions for the implementation of new procedures. From them, one of the most recent agent methodologies, named Prometheus, has been selected to integrate the system specifications, inter-agent coordination protocols and agent inner processes as components of an ATM conceptual model, described in section 5. In order to simplify the applicability of the mentioned methodological approach, it has been applied for analysing and modelling a

particular air traffic scenario of arrival air traffic operations. In this case we will consider this particular ATM model as an Air Traffic System (ATS). Moreover the model is focused on the inner architecture of an ATC System. Although these simplifications, the obtained model under this methodological approach can be extended for adding new gate-to-gate air traffic scenarios, coordination procedures as well as improved versions of the technical support to execute mentioned procedures. Section 6 resents guidelines for a software implementation and results of an illustrative example of current implementation state. Finally, conclusions are presented in section 7.

2. Simulation and design analysis tools for new air-traffic concepts

The simulation of air traffic scenarios with different levels of fidelity is the mainstay of the methodology used widely by the scientific community to develop new operational concepts [8-9]. Real-time simulations are, in general, HITL simulations intended for human factors evaluation while navigation and/or traffic control procedures are performed. Fast-time simulations are centred on the analysis of several issues no specifically focused on human factors: mathematical models, algorithms, negotiation and/or decision-making processes, quality of service measures, etc. Obviously real-time simulation platforms are able to carry out fast-time simulations by including computer models for performing tasks assigned to the human element.

Some of the multi-aircraft simulators currently used for air traffic research purposes are: NLR's Air Traffic Control Research Simulator (NARSIM)) [10], Pseudo Aircraft System (PAS) [11], Target Generation Facility (TGF [12], ATC Interactive for the future of air traffic control[13] and Multi-aircraft Control System (MACS) [14]. Three main consequences can be derived from the analysis of previous works. First, more detailed proposals are necessary for supporting automated air-ground coordination processes. Second, modelling such automatic coordination procedures system requires a parallel specification of their associated systems (i.e. user interfaces, mathematical models and algorithm for making decisions, etc). Third, new conceptual models and simulation frameworks require a high modularity and scalability for making possible the progressive incorporation of new or modified procedures and their associated systems as they are designed or evaluated.

In the context of modelling such complex distributed systems, Multi-Agent Systems (MAS) theory gives natural solutions [17-18]. This theory provides a suitable framework to analyse and model the organization of a set of autonomous ATM entities that coordinate and negotiate their actions in order to achieve their respective goals.

3. Multiagent approaches for future air traffic scenarios

The dynamic nature of air traffic and its geographical and functional distribution have attracted the attention of agent researchers since the last decade. For example, Optimal Aircraft Sequencing using Intelligent Scheduling (OASIS) [1] is a system used at the airport of Sydney to help air traffic controllers on arrival and approach air traffic sequencing

operations. However, most of agent-based contributions are oriented to developing very basic aspects of the ATS system. The proposals can be classified into three categories:

i. Analysis of negotiation patterns between agents in free flight air traffic scenarios. Within this category, Wangerman and Stengel analyse the dynamic behaviour of aircraft in the airspace as an intelligent system (Intelligent Aircraft/ Airspace System or IAAS) [19-21]. This system consists of three types of agents, airlines, aircraft and traffic managers, and the dynamic behaviour of the agents of IAAS is analysed from the perspective of a distributed approach. In this context, the principles of negotiations are a way to implement distributed iterative optimization of IAAS operations. In [22] Harpert et al. propose an agent-oriented model of an ATM in a free flight and a model of distributed decision making for the resolution of conflicts in the ATM. This proposal includes a distributed optimization scheme in which the agents generate and evaluate proposals options that best suit their preferences on a utility function and a multi-attribute decision tree scheme. Besides, the declarative capabilities of an intelligent agent can be modelled by expert systems [23-25]. To set the ground rules of the expert system, tasks were classified in [21] into four groups: emergency tasks, tasks of a specific mode of operation, negotiation tasks and routine tasks.
ii. Avionics systems for autonomous operations in distributed air traffic management scenarios. Works in this category propose designing avionics systems based on multi-agent systems. The proposed developments basically consist of systems for automatic conflict resolution, automatic warnings and recommendations to the crew [26-28]. In [26] authors propose a design of intelligent traffic agent developed to detect and solve conflicts on board in a free flight environment. An extension of previous work presented in [27] proposes a design of an executive agent that resolves conflicts for both the traffic and bad weather areas. In this case an agent's hierarchical architecture is suggested for making decisions based on the information produced by a traffic agent proposed by [26] and a weather agent. Finally, in [29] capabilities required of future flight management systems (FMS) in the cabin were characterized by means of agent-oriented analysis of the air navigation and arrival operations into a distributed environment. This analysis was later extended to define capabilities required to carry out automated arrivals and departures at uncontrolled airports [30].
iii. Simulation systems for the analysis of advanced air traffic concepts. This group includes several simulation platforms used for the design and validation of procedures and systems proposed for next generation of air traffic scenarios. In [15, 17] a multi-aircraft control platform is proposed to increase the realism and flexibility of HITL simulations. Functional descriptions of pilot and ATC perspectives within this platform are presented in [15]. In [17] the ATC agent model is analysed in order to identify its roles and responsibilities in future ATM systems. Another development of functional architecture airspace for an Airspace Concept Evaluation System (ACES) is presented in [16]. In a later work an agent-oriented model of the CNS/ATM infrastructure of ACES was proposed in [31]. Finally, in [32] a design of an experimental air traffic simulator implemented as a Java environment SMA is presented.

The proposals based on multi-agent systems presented above cover a wide spectrum of issues related to air traffic operations in a free flight environment. However it is clear that effective implementation of the new operational concepts involved in the future scenario still requires more structured models that take into account the tight relationship between procedures and systems to support them. In addition, as it was explained before, the model should be scalable enough to allow a progressive integration of the following elements into the model:

- Operating procedures for ATC and aircraft, specifying: (i) roles and tasks assigned to ATC and flight crew and (ii) coordination rules and negotiation protocols among the involved agents
- ATC and on-board functionalities to help to execute such procedures at several automation levels.
- Underlying mathematical models and algorithms to give support to previous functionalities.
- High level languages that allow for accurate intercommunication between aircraft and ground systems.

Fortunately, methodologies for developing multi-agent systems have reached a noticeable degree of maturity in recent years, becoming an invaluable tool to achieve a comprehensive analysis and modelling of complex scenarios. Thus, the analysis and modeling of interactions in terms of coordination and negotiation strategies between agents provides useful guideline for developing new schemes for developing automated ATC and navigation procedures. In turn, the study of the agents' behaviour and their internal architecture for the mentioned coordination processes provides a more precise identification of the functionalities required by on-board and ground systems to execute mentioned these procedures.

4. Current methodologies and tools for analysing and designing MASs

Agent methodologies provide with a set of guidelines to facilitate the development of multi-agent systems over several stages since the initial draft of idea until the final detailed design. In this way, current multi-agent technology provides practical and formal methodologies to analyse and design, in a structured and consistent manner, the following issues: (i) roles and functionalities of autonomous entities (agents) that take part in an operational scenario, (ii) interactions between agents (or agent protocols) and (iii) inner architecture and dynamic behaviour (processes) of agents. Besides several agent platforms have been proposed as middleware tools for translation the conceptual model into an executable model.

The design of MAS requires not only new models but also the identification of the software abstractions, since this paradigm introduces a higher abstraction level when compared to traditional approaches. They may be used by software developers to more naturally understand, model and develop an important class of complex distributed systems. The key aspects of problems being modelled under a MAS methodology are: establishing a set of

coarse-grained computational systems (agents) and interaction mechanisms for a goal to obtain in the system that maximizes some global quality measure, assuming a certain organizational structure which can be assumed to keep fixed (agents have certain roles and abilities that do not change in time).

Some of the main agent-oriented methodologies are MASCommonKADS [33], Tropos [34], Zeus [35], MaSE [36], GAIA [37], INGENIAS [38]. In [30-41] a comparative analysis of the main methodologies is presented. MAS-CommonKADS is a methodology for knowledge-based system that defines different models (agent model, task model, organizational model etc.) in the system life-cycle using oriented-object techniques and protocol engineering techniques. Tropos is a requirement-based methodology; Zeus provides an agent platform which facilitates the rapid development of collaborative agent applications; MaSE is an object-oriented methodology that considers agents as objects with capabilities of coordination and conversation with automatic generation of code and Unified Modelling Language (UML) notation; Gaia is intended to go systematically from a statement of requirements to a design sufficiently detailed for implementation; and INGENIAS proposes a language for multi-agent system specification and its integration in the lifecycle, as well as it provides a collection of tools for modelling, verifying and generate agents' code.

A descriptive analysis these methodologies are beyond of these chapter goals. However after a previous study we have selected Prometheus methodology as the most suitable one. Prometheus agent-oriented well-established methodology has been selected to provide guidelines to develop the mentioned multi-agent system [42]. We argue that Prometheus suits well for solving our problem due to: (i) the highly detailed guidelines for defining the initial system specification, (ii) the modularity of the agent's internal architecture around the concept of capability (providing a direct correspondence between capabilities and functionalities of airborne and ground systems), (iii) the easy translation from the conceptual model into an executable model by means of current agent platforms that provides software infrastructure as it will be explained later on.

Prometheus methodology covers the entire process of design and implementation of intelligent agents. It includes three phases (see Figure 2): system specification, architecture design and detailed design [42].

System Specification stage defines the objectives or goals of the system. Goals help to identify functionality required to achieve them, as well as a description of the interface between the system and its environment in terms of inputs (*Perceptions*) and outputs (*actions*) of the system. The identification and refinement of the objectives are carried out, in an iterative manner, from the definition of different use case scenarios. Scenarios illustrate the operation mode of said system. The concept of scenario (or use case scenario) comes from the object-oriented software methodologies, but with a slightly adapted structure that provides a more integrated than the mere analysis of the isolated system.

Later on, in the Architecture Design phase, the description of the system structure is represented by means of diagrams that identify the agents of the systems and their interactions in terms of communication messages and protocols. Protocols represent specific communication schemes. They can be modelled using Agent Unified Modelling Language (AUML) that describes the interactions of agents in different scenarios of use cases.

Finally, the Detailed Design phase consists of designing internal processes carried out by each agent and an inner architecture that describes how these processes are organized and implemented. Prometheus proposes implementing agent tasks by means a set of different plans that are triggered when determinate events occurs. Plans are grouped into several groups associated to the execution of specific tasks. A group of plans as well data used or produced by them constitutes an agent capability. Then focus of this stage is to define capabilities, internal events, plans and detailed data structures. In this way capabilities are modules within the agent that use or provide related data types. Capabilities can be nested within other skills so that in the detailed design, the agent will have an arbitrary number of layers in an understandable complexity at each level. Thus, capabilities can be constituted by other sub-capacities or, at lowest level, by plans, events and data. The plans set out the set of tasks performed to achieve a particular purpose. They are also triggered by certain events (internal or external messages, perceptions, etc.). As a result of this stage are general diagrams of each of the agents (which show higher level capabilities of the agent), charts of capabilities, descriptors detailed plans and data descriptors.

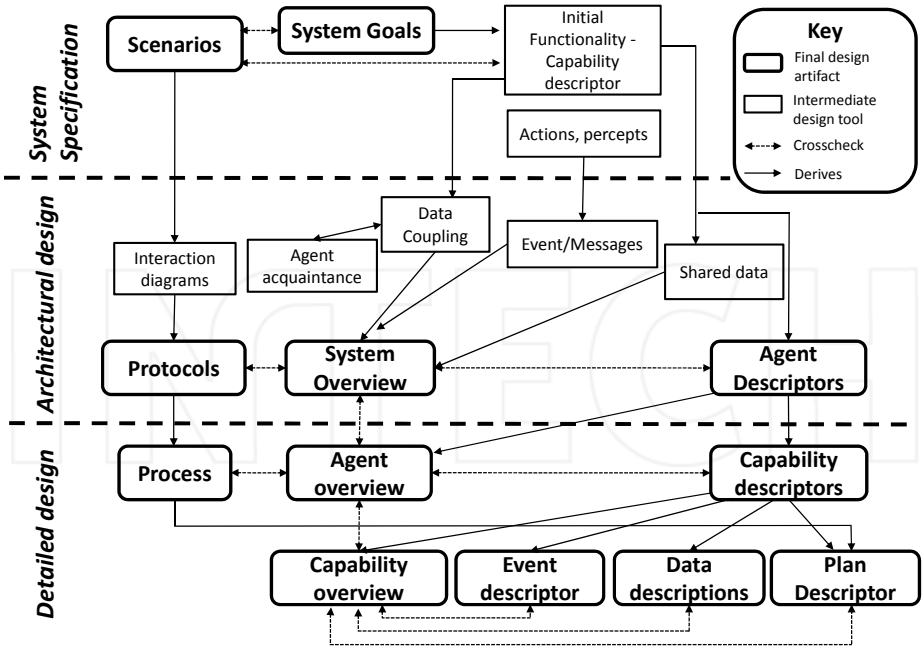

Figure 2. Prometheus methodology phases [42]

Moreover, the tool Prometheus Design Tool (PDT) facilitates the tasks of the developer over the previous stages to provide information about possible inconsistencies in the design [43]. In addition, several software tools have been developed in recent years for software implementation of system multi-agents. One of the most extended is the Java *Agent DEvelopment Framework (JADE)* Platform [44]. JADE simplifies the implementation of multi-agent systems through a middleware that provides several resources through a set of library classes aimed to:

a. Implement agent's tasks into a several JAVA classes named behaviours.
b. Provide agent intercommunication that complies with the *FIPA (Foundation for Intelligent physical Agents)* specifications [45].
c. Supply services for create and manage cycle of live of agents as well are their services into the multi-agent system.

JADE behaviours can be classified onto simple behaviours and composite behaviours. In turn, simple behaviours can be classified as:

a. One-shot behaviour, an atomic task to be carried out once, used here for initialization tasks;
b. Cyclic behaviour, which is iterated while exists, such as messages listening and processing;
c. Waker behaviour, or a one-shot behaviour invoked after a certain time; and
d. Ticker behaviour or a cyclic behaviour which performs a series of instructions executed keeping a certain fixed time, used in the platform for simulation numeric computation and graphical output.

Composite behaviours are three:

a. FSMBehaviour that consists of a class that allow defining a Finite State Machine by means sub-behaviours, where each of them represents an machine state
b. SequentialBehaviour that executes its sub-behaviours in a sequential way, and
c. ParalellBehaviour that executes their sub-behaviours concurrently and ends when a certain condition is satisfied (for one, several or all of them). In this way, agents are able to concurrently to carry out different tasks and to keep simultaneous conversations.

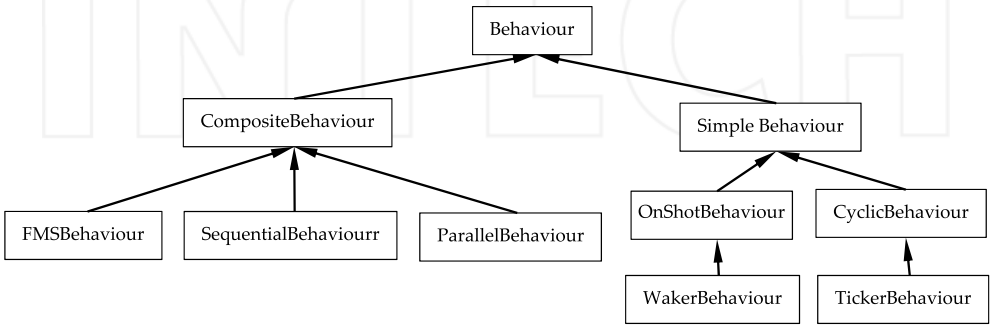

Figure 3. JADE Behaviours

5. Appling Prometheus methodology for designing an arrival TBO conceptual model

As explained in the previous section, Prometheus methodology carries out an iterative process on three phases: specification system, architecture design and detailed design. Each of these phases provides guidelines for designing several model components. These components produce a hierarchical structuring mechanism which makes possible a model description at multiple levels of abstraction [19]. In addition, the structured nature of design components facilitates crosschecking for completeness and consistency of the model in each design phase.

5.1. System specification

In this phase, goals of our ATS model are identified. In turns goals are captured from a set of *scenarios* that illustrate essential aspects of the system operation. Scenarios and goals help to recognize initial system functionalities and to examine the system-environment interface in terms of inputs *(Perceptions)* and outputs *(actions) [45].* Thus, scenarios are use cases that contain a sequence of steps each of them relating to a goal, an action, a perception or another scenario.

For outlining the mentioned scenarios, a generic automated air traffic scenario was considered as a distributed process where several autonomous and proactive entities (agents) plan and execute a set of coordinated tasks to provide arrival and approach free of conflict 4D trajectory. This operational scenario is particularly critical in arrival air traffic operations due to the high variability of the speed, heading and altitudes that could affect to the degree of predictability of several converging trajectories. Moreover, guidelines from scenario proposed in DAG-TM (CE-11) project [46] have been taken into account. According to referred guidelines the flight crew: *(i)* could negotiate arrival preferred trajectories with the ATC; *(ii)* is responsible for maintaining longitudinal spacing between consecutive aircraft once a trajectory (o constraints) has been assigned.

In this operational scenario, the following agents have been identified: Aircraft, Air Traffic Control (ATC), Meteorological Service Provider (MPS), Airspace Resources Provider (ASP) and Airline Operational Control (AOC). In addition several ATC agents could be defined in order to coordinate arrival ATC tasks with the ATC en-route or departure ones. MSP, ASP and AOC agents' functionalities have been used to define the information required by the ATC and aircraft agents as well as essential protocols to accomplish this information.

Use case scenarios have been selected and organized taking into account perspective that each agent have about the generic air traffic scenario. Then, five root scenarios have been defined: *(i)* Manage Aircraft, *(ii)* Manage ATC, *(iii)* Manage Airline Operational Control, *(iv)* Provide Airspace Resources *(v)* Provide Weather Information.

Tasks of each one of above scenarios have been grouped into new sub-scenarios and so on. Figure 4 depicts a list of the most significant sub-scenarios deployed from the previous one. In particular *Manage navigation procedure* scenario and *Manage ATC* scenario are developed until the lowest level. In addition, this scenario architecture shows that air-ground negotiation processes are contained into specific air-ground negotiation scenarios which are shared by *Manage Aircraft* and *Manage ATC* scenarios.

To illustrate how scenarios can be deployed we will focus on the *Manage ATC scenario*. This scenario contains the following four scenarios:

- *Update ATC environmental information scenario,* which covers associated processes to collect information about status and intentions of aircraft, airspace resources (including restricted flight areas), weather conditions, etc.
- *Manage ATC procedures scenario*. It includes the processes related to maintaining the separation of aircraft to achieve an efficient traffic flow.
- *Manage on board surveillance scenario*. This scenario contains tasks for monitoring air traffic. These tasks are aimed at identifying aircraft trajectory deviations and potential conflicts with other aircraft or obstacles. It also provides viable solutions for correcting these anomalies and events for triggering specific processes in order to implement the solutions mentioned above.

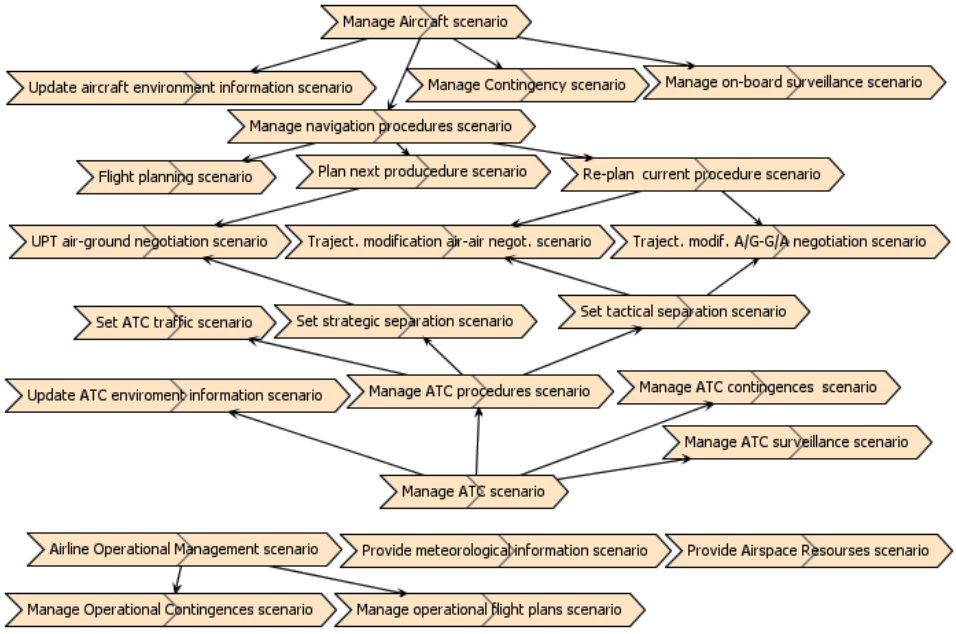

Figure 4. Architecture of the main scenarios for Trajectory Based Operations

- *Manage contingences scenario* that includes tasks for analysing air traffic contingencies and circumstances in which they occur (e.g. aircraft malfunctions, on-board contingences, etc.). It also includes decision-making processes to determine actions to be carried out regarding the management of traffic control procedures.

Focusing on the *Manage ATC procedures scenario*, the next three scenarios are deployed:

- *Set ATC traffic* involves actions for *receiving* or transferring air traffic from or to other adjacent ATC agents.
- *Set strategic separation*. This scenario contains tasks for planning aircraft trajectories and assigning them by means a negotiation process. *Therefore* it is a key scenario for modelling automated procedures for TBOs and it should contain several negotiations sub-scenarios.
- *Set tactical separation*. This scenario contains tasks for modifying current flight trajectories when unforeseen contingencies arise. The tasks performed in this scenario are twofold: (i) to provide specific instructions for activating protocols aimed at aircraft separation in extreme situations of short-range conflicts and (ii) to authorize and supervise air-air negotiation for self-separation when separation responsibility has been delegated on the aircraft.

From the above scenario architecture a goals tree can be obtained. Lowest level goals help to identify functionalities and processes that the agent has to carry out to achieve them. For example, Figure 5 shows a particular set or goals that results from the *ATC manage* scenario. On it, the goal *UPT air-ground negotiation* consists of several sub-goals such as generate proposals for aircraft, evaluate proposals from aircraft or establish and an air-ground communication protocol for negotiate mentioned proposals and counterproposals. In addition, functionalities help to identify actions, perceps and data used or generated by the agents. Then, for the ATC agent the following perceptions can be identified:

- Perceptions form external sensors: data from radar systems, WAN receivers, etc.
- Perceptions from human-machine interface: menus and inputs options.
- Actions: display traffic data and ATC procedure state data on screams.

5.2. Architecture design and negotiation protocols

In the architecture design phase the following aspects of the overall system are defined:

- The system *overview diagram*. This diagram represents the static structure of the system, tying agents and main data used by them as well as their perceptions and actions. Furthermore communication interactions between agents are considered.
- The set of *interaction protocols* that capture timing of communication of related messages between agents. These protocols are derived from the scenarios defined in the specification phase protocols and, therefore, they describe the system dynamic behaviour. Then, they have been depicted using an AUML notation [47] .

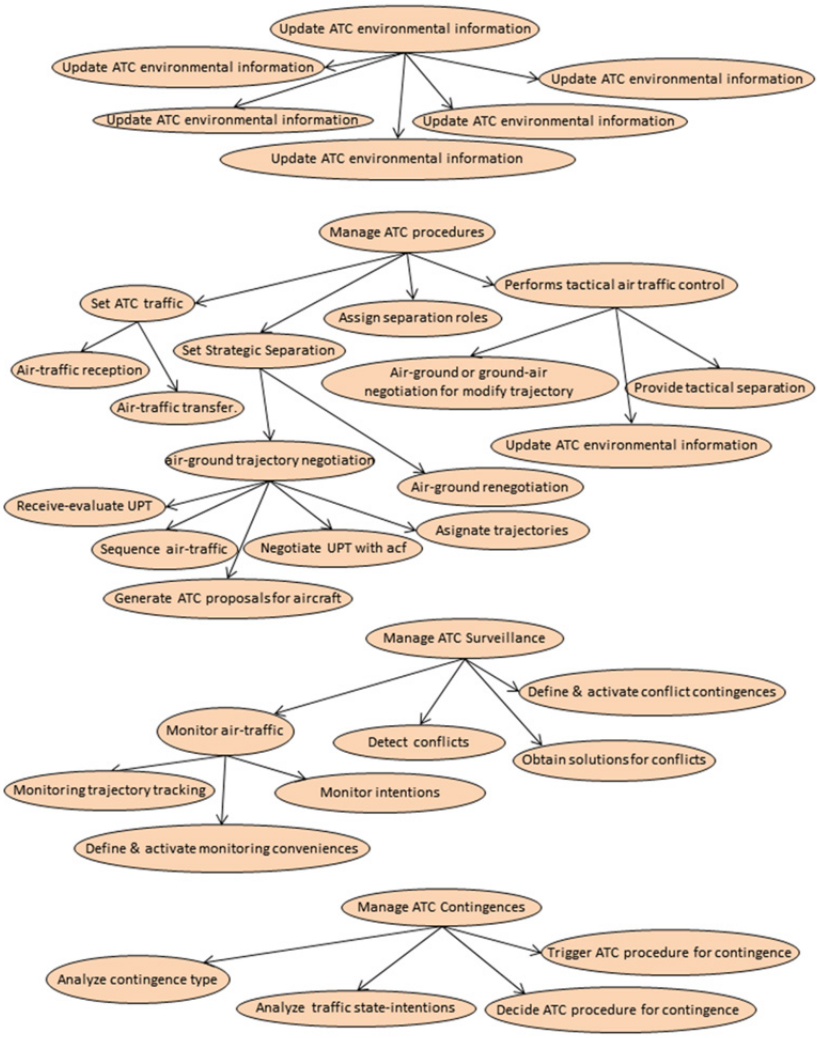

Figure 5. ATC agent goals

Figure 6 shows a simplified representation of the system overview diagram. The simplification consists on representing only the main actions and perceptions of the ATC and aircraft agents as well as the main communication protocols. In a more complex system overview diagram, all these elements should be signed for all the agents.

After identifying the interaction protocols in the system overview diagram, protocols are designed. For the ATC and aircraft agents, protocols are aimed to: (i) improve agents' knowledge base about the environment and/or the other agent's intentions, (i) negotiate trajectories that could be in conflicts.

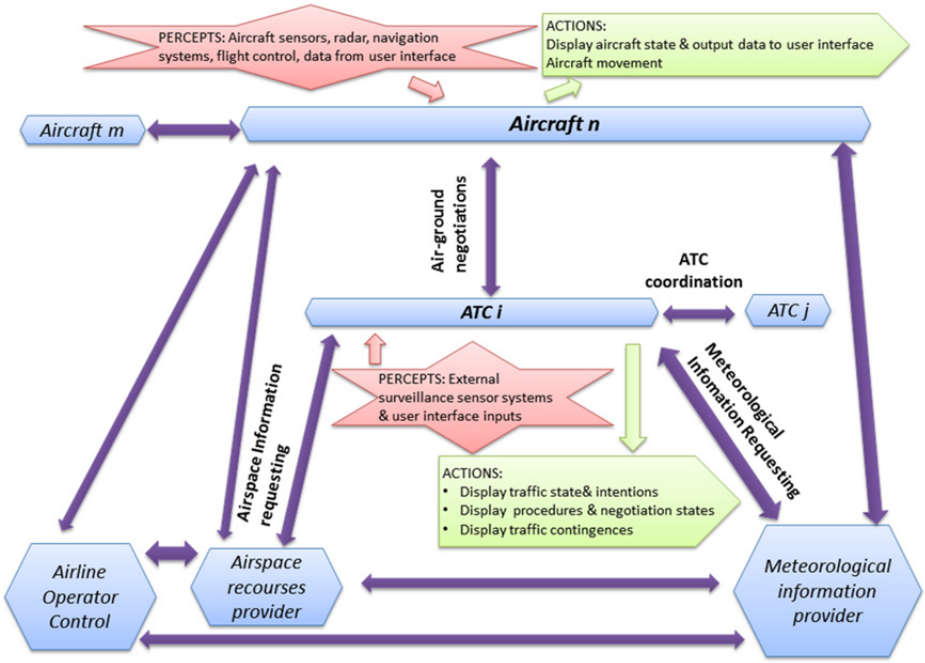

Figure 6. Simplified architecture overview

For a better understanding of automated air-ground coordination aspects, we will focus on describing a proposed air-ground negotiation protocol (see Figure 7). This protocol represents the core of the both the ATC strategic planning tasks and the aircraft navigation planning tasks. Although new scheme of negotiation can be defined from this design, all of them will use similar functionalities to evaluate proposals and generate counter-proposals. Therefore, this protocol and its associated functionalities provide guidelines and specification enough for developing new aircraft and ATC coordination procedures.

In Figure 7, the on board computation processes are represented on the left side of the aircraft agent lifeline. On the right side of the ATC agent lifeline we can observe computation performed by ground systems. Moreover, on the right side of this ground computation system, a new lifeline for other aircraft agents is showed.

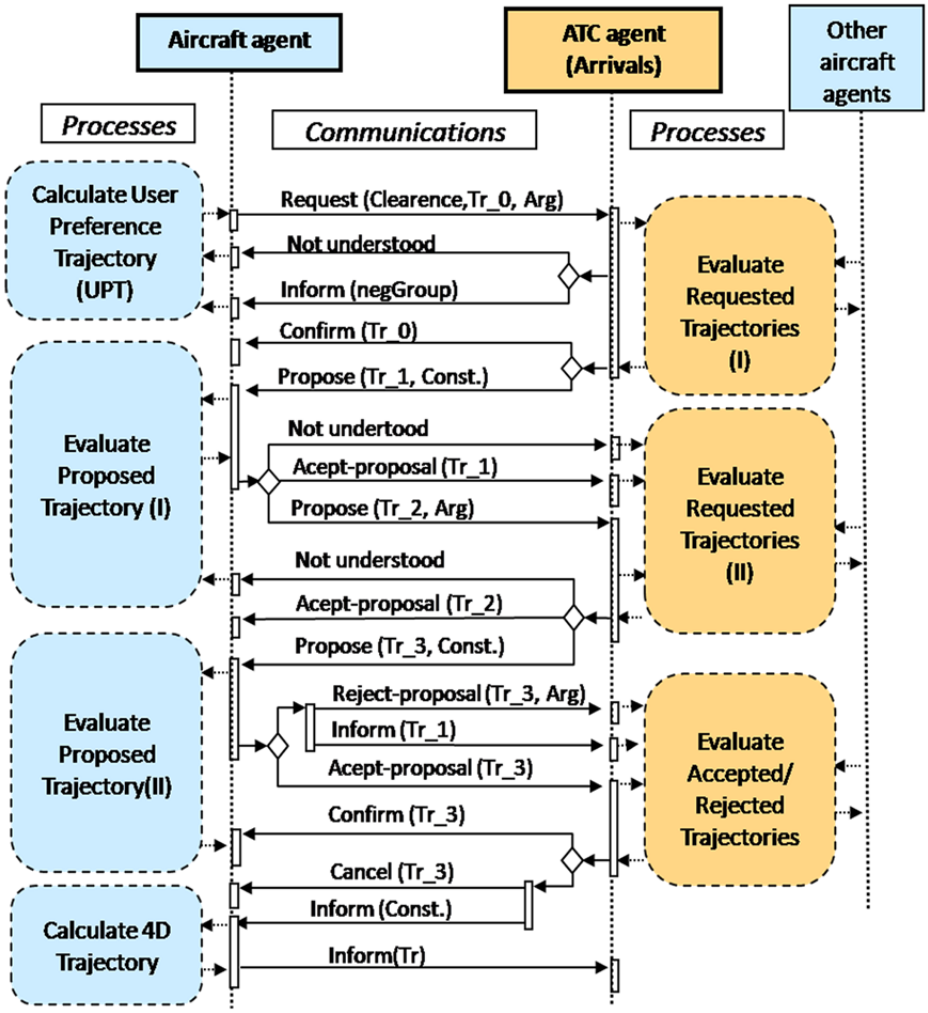

Figure 7. Air-ground Negotiation Protocol

The proposed arrival negotiation protocol can be divided into two phases that are described next:

a. Phase 1: Before reaching Time Limit for Requesting Trajectory (TLRT)

In this phase aircraft calculates their preferred 4D Trajectory (Traj_0). To perform this computation, each aircraft agent uses the available information about meteorological conditions and arrival routes. This information is obtained by means of a previous communication procedure (no represented in Fig. 7) with the Meteorological Information Provider agent and the Air-space Resource Provider agent. Once the 4D trajectory is calculated, the aircraft requires clearance to the ATC to execute it. In this case, the TLRT represents a deadline time for requesting mentioned clearance.

The ATC agent receives *Request messages* from different aircraft that are periodically processed in-batch. After receiving these messages, the ATC evaluate if requested trajectories are free of conflicts. As a result, the ATC could confirm the trajectory initially preferred by the aircraft or it could propose new constrains for a new one *(Traj_1)*. If the aircraft is agreed with previous information, it sends a corresponding message and the communication process finalizes. But if the aircraft wish to flight an alternative trajectory (i.e. a faster one), the negotiation protocols continues in a second phase.

b. Phase 2: Call For Proposal

In this phase, the aircraft makes a second counter-proposal in order to improve the previous ATC proposal, arguing reasons for it (for example certain operational contingences). These kinds of proposals *(Traj_2)* will be evaluated by the ATC. Those one that can be feasible will be accepted. In other case, the ATC will perform a new proposal *(Traj_3)* that the aircraft in turn can refuse or accept. When an aircraft refuses mentioned ATC proposal, it will have to select one that satisfy previous ATC constrains. But if the aircraft accepts cited ATC proposal, it will have to wait an ATC confirmation message before implement such proposal. This confirmation is necessary due to the ATC has to analyze air traffic state after receiving several aircraft messages accepting or refusing *Traj_3* proposals. Then the protocol ends with aircraft messages informing about details of the last cleared and accepted trajectory.

Finally note that the software implementation of messages used in this protocol can be performed by a normalized FIPA support [45].

5.3. Detailed design

Finally, in the detailed design phase, the dynamic behaviour and the internal agent architecture are projected. The dynamic behaviour is described by a set of processes that agents carry out when they interact or make decisions. The internal agent architecture is represent by means an agent overview diagram that shows how these processes are organized.

Processes are represented by a flow diagram that links protocol messages with internal functionalities that evaluate and generate proposals. Notation used for depicting processes is showed in Figure 8. This notation is slightly different to the UML notation so that, instead of focusing on the activities it is focused on communications. Besides, we extend notation proposed by Prometheus methodology in order to include information about the different states of the air-ground negotiation protocol. These negotiating states are intended to enable automated negotiation processes whose evolution can be understood in supervisory tasks of pilots and controllers. Figure 9 shows the process carried out by the ATC while the air-ground negotiation protocol, previously presented, is in progress.

Agent processes like the described above can be implemented by means of plans. Then plans contain a set of instructions in order to: (i) carry out computations, (ii) take decisions (iii) generate or receive messages and new events. Moreover plans are to be triggered by specific events such as arrival messages or events generated by other plans.

The agent diagram overview consists of an agent architecture representation that indicates how all these plans are organized. Therefore, it shows interaction between plans, shared data and events. In addition Prometheus methodology proposes to organize plans that share similar functionalities and data into capabilities. Figure 10 represents Prometheus notation for representing elements of the agent overview diagram. Then, Figure 11 represents the ATC agent architecture diagram overview. On it, main capabilities of the ATC agent are depicted together with data used or produced, agent inner events and communication messages.

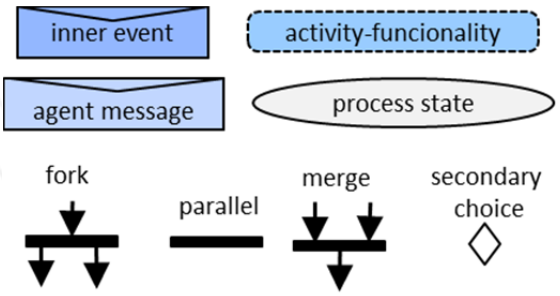

Figure 8. Notation used wihin agent processes

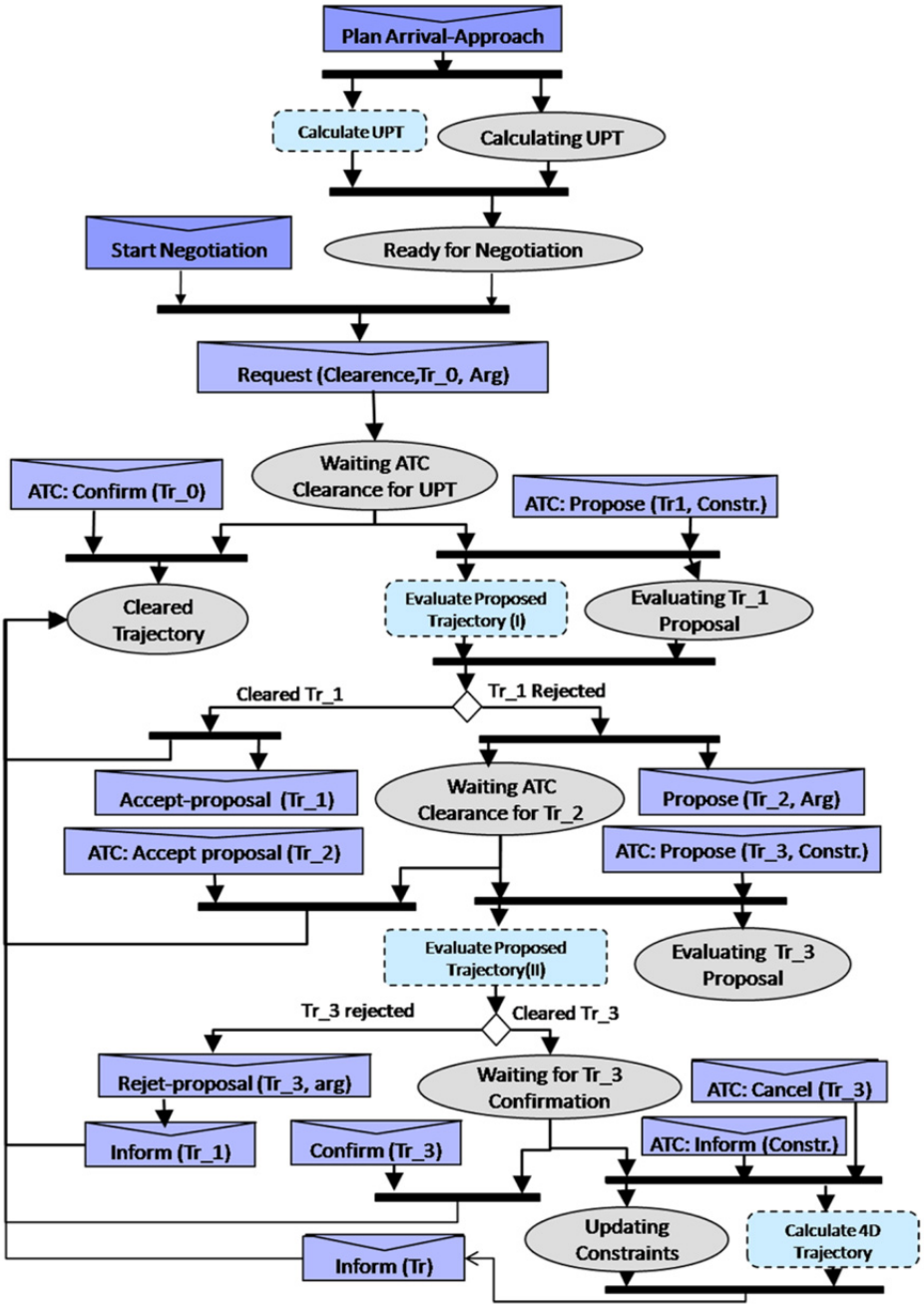

Figure 9. ATC Process Diagram for Air-Ground Negotiation Protocols

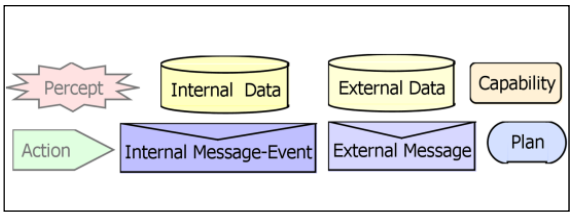

Figure 10. Prometheus notatino used in agent and capability overview diagrams

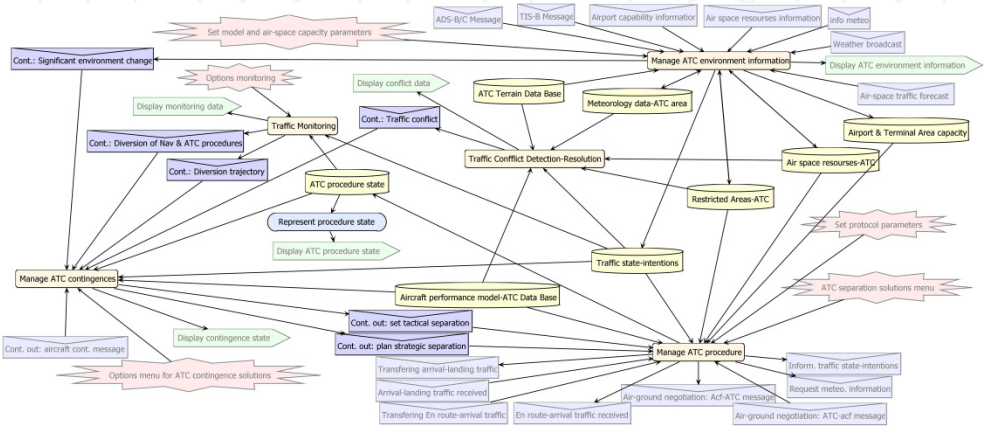

Figure 11. ATC agent architecture

Taking into account scenarios, functionalities, processes, events and data established in previous phases, plans has been grouped into the following capabilities:

- *Manage ATC Environment Information:* This capability is associated with the goal of maintaining an updated on-board environmental knowledge. Plans for this capability capture information about weather forecast, restricted areas, air space recourses (e.g. available arrival routes and gateways), air space contingency events concerning to significant environmental changes and aircraft contingence events.
- *Traffic Monitoring:* This capability checks the state and intentions of the aircraft according to air-ground agreements previously negotiated. In case of divergences, an air-traffic contingence event is generated informing about it.
- *Traffic Conflict Detection-Resolution:* As its name suggests, it is responsible for detecting conflicts with other aircraft or obstacles (terrain, adverse weather areas, etc.). It also provides a set of ranked proposals for conflict resolutions. Furthermore, proposals are negotiated and/or implemented by means of other capabilities. To achieve above goals, plans of this capability are grouped into two sub-capabilities: *(i) Conflict Detection Capability and (ii) Initial Conflict Solution Capability*.

Conflict Detection Capability contains plans to implement algorithms for conflict detection. Therefore, it can be constituted by several plans each of them contains a specific model to detect short, medium and long term conflicts. Plan inputs are data about predicted trajectory, restricted areas, surrounding traffic state and intentions. Plans of this capability are triggered by events generated by plans of other capabilities that perform surveillance tasks as well as testing tasks within the trajectory planning processes. Conflict data calculated by previous plans are used by a specific plan to obtain a detailed conflict description and to generate conflict events.

Initial Conflict Solution Capability uses several inner plans to supply solutions according conflict data input. Results of these plans are used by other ones that generate conflict contingence events. Then events also contain associated information about feasible conflict solutions.

- *Manage ATC Contingency.* This capability deal with deciding which kind of ATC procedural tasks are to be carrying out according to the information contained on received contingency events. To make decisions, plans of this capability take also into account current states and intentions of aircraft traffic. Information about the procedural tasks that have to be performed are included into contingency output events that will trigger specific plans to execute such tasks. These plans are grouped into the ATC Procedures Management capability that is described next.
- *Manage ATC Procedures*. Plans of this capability carry out strategic and tactical actions aimed to maintain aircraft separation. These plans are grouped into the next for sub-capabilities:
  - *Implementation of ATC procedures*. This capability has a first plan that take into account air traffic conditions to generate events that trigger plans for: (i) ATC coordination in order to receipt o transfer air traffic, (ii) planning and assigning trajectories, (iii) establishing point for initiate air-ground negotiations and (iv) assuming or delegating aircraft separation responsibilities.
  - *Strategic Separation*. This capability is modelled through two basic plans. One of them drives processes of trajectories negotiation. The second one manages other supplemental plans that implement re-negotiation processes of trajectories previously assigned to a group of aircraft and pending of execution when a contingency arise.
  - *Tactical Separation*. Plans of this capability manage processes triggered by contingencies that require this type of action (e.g. separation loss contingency when air traffic flows converge). Obviously, in the context of TBO, tactical actions should be reduced to: (i) delegate or regain the separation control role depending as a function of the air traffic state and other contingences and (ii) enable separation control protocols in extreme short-range conflicts.
  - *ATC Coordination.* This capability includes plans to coordinate for air traffic transferring between adjacent ATC units. Events that trigger these plans come from the sub-capacity ATC procedure execution. The detailed design of this capability includes a specific plan to implement ATC coordination protocols with other adjacent units.

6. Implementation

Descriptors and diagrams of the components in the described conceptual model contain all the necessary information to carry out implementation. However, not all the components obtained in the three phases of the methodology have to be implemented. The executable model consists of the entities that have been developed in the detailed design phase (i.e. agents, capabilities, plans, data, events and messages).

As it was explained in section 4, Prometheus methodology provides a full life-cycle support tool (PDT tool) to develop multi-agent systems. Current version of PDT provides support for: *(i)* designing most the design artefacts within the Prometheus methodology, *(ii)* cross-checking for consistency and completeness for the conceptual model, *(iii)* automatic generation of skeleton code in JACK agent-oriented programming language [48].

The conceptual model detailed above is currently at implementation phase. Although facilities of automatic code generation of PDT, we have opted for using the JADE Platform [39], cited in section 4, due to: *(i)* it is one of most extended multi-agent platforms and, *(ii)* it provides the FIPA standards [49] infrastructure for inter-agent communications and for managing software agents distributed across multiples hosts. As it was explained, architecture of JADE agent is built upon the behaviour concept rather than a plans-based architecture. Then agent plans can be generally implemented into JADE behaviours in a quite straightforward way.

On the other hand, continuous simulation requires, in nature, implementing the aircraft dynamic over a continuous-time model. It is essential to carry out real-time and human-in-the-loop simulations in order to analyse in detail and validate the design accordingly to the expected global behaviour. Also it is suitable for fast analytical simulations intended for preliminary designing and evaluation of cockpit systems and underlying mathematical models and algorithms (e.g. for 4D-trajectory guidance, conflicts detection and resolution, etc.). However, while the detailed models are not available, the proposed conceptual model enables discrete event simulation. In this case, events can be generated by random functions implemented within capabilities plans representing underlying models as *black boxes*. In this way, for an initial implementation phase, random functions to generate events are implemented into agent plans. In a later phase, the executable model can be refined when functions are replaced by specific underlying models as they are developed.

6.1. ATC model implementation under JADE platform

Figure 12 illustrates an adaptation of the ATC agent capacities-based architecture to a behaviours-based one using JADE behaviours described in section 4. Each one of the agent capabilities has been defined as behaviours that run in a parallel way. Previous behaviours could be progressively broken down into new behaviours, so that, at lower-level, behaviours correspond to plans of the conceptual model.

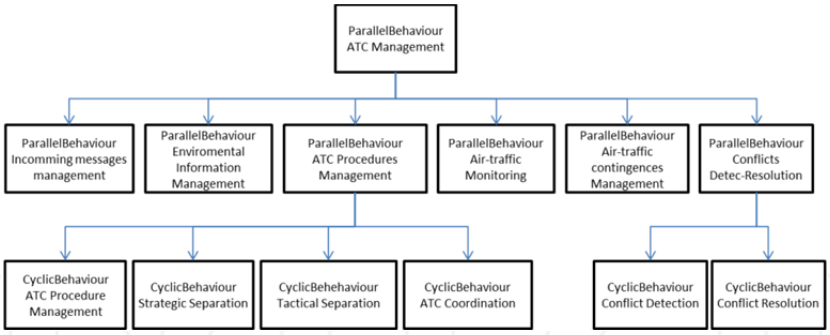

Figure 12. Figure 12. ATC agent architecture based on JADE Behaviours

6.2. Dynamic prototype example: An experimental air traffic simulator

As example of a JADE implementation we summarize the architecture of an Experimental Air Traffic Simulator (EATS) [32] that we have developed under a JADE support. This simulator includes agents described in the last section as well as other two agents with particular purposes into a simulation environment: the *Configuration Agent* and the *Pseudopilot Agent*.

The *Configuration Agent* is required to define the set of initial simulation parameters (e.g. aircraft type, available routes, etc. The *Pseudopilot* Agent has been designed with a twofold purpose. First, it is a desk control that allows to an unique pilot-user (named pseudopilot) to have control over several aircraft. Second, it represents a graphical display, providing significant information about the state and intentions of surrounding traffic for each selected aircraft. This interface has been implemented as a separated agent (and not like an aircraft agent component), to centralize in a unique interface the access to each aircraft. Besides, it plays an important role (especially in the near future scenarios) to design and evaluate specific on-board man-machine interfaces like the CDTI cockpit display [50]. The CDTI allows seeing the surrounding traffic and, what is more relevant, the intentions of the surrounding aircraft. To access to a particular aircraft, a mouse click over the icon symbol is required. Once the aircraft is selected, it is placed at the central position of the *pseudopilot view window*, and the movement and the position of other aircraft are represented in relation to it. At the same time, the control window of the selected aircraft will be opened.

Then air-traffic controllers and pilots can interact with agents by means of two types of consoles. In one of them an Air Traffic Controller can monitor the positions of different aircraft and send several data instructions to a specific aircraft. In the other one a user pseudopilot that receives orders from the ATC (via voice or via data messages), carries out the necessary actions to fly the aircraft according to these orders. Besides, pseudopilot agent can be configured to automatically execute ATC mentioned data instructions.

Figure 13 shows screenshot of this application. It represents a view of the ATC interface constituted by and screen for displaying the traffic and a window console for interchanging data and instructions with a particular aircraft. Besides, in the same screenshot two aircraft control windows (A320 and Cessna) are deployed.

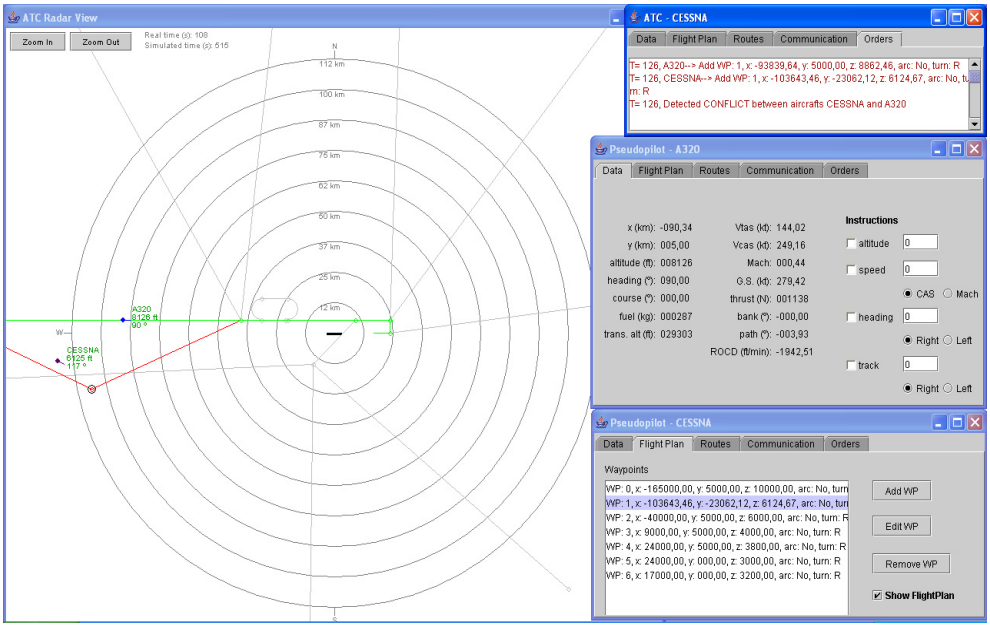

Figure 13. Application screenshot for the described scenario

To carry out communications between agents, a Communication class with specific methods has been designed. In particular, the air-ground communication between the aircraft agents and the ATC agent is carried out with the following messages:

a. Messages sent by the aircraft to the ATC: message to inform about the state vector and planed route, message to inform about the possible modification of the altitudes of the flight plan to initiate a continuous descent approach to the airport.

b. *Messages received in the aircraft from the ATC*: instruction messages (changing altitude, heading, speed, a flight plan waypoint, etc.) and messages of conflict detection with other aircraft. Starting from this nucleus of air-ground communications, new types of messages can be implemented in future EATS extensions with the purpose of establishing more complex negotiations between aircraft and ATC.

c. *Messages sent by the aircraft to others aircraft*: message to inform about the state vector. This air-air communication provides information to each aircraft about its surrounding air traffic.

Besides the previous communications, there are other communications involving the Meteorology Information Provider agent (to obtain atmospheric information) and the Airspace Recourse Provider agent (to request the available routes). Moreover, the communication between the aircraft agent and the pseudo pilot agent represents the communication between a physicals agent (the aircraft) and a man-machine interface like the CDTI.

Apart from above infrastructure for agent communications, the current prototype version implements the aircraft aerodynamic and simple agent making-decision mechanisms. Thus, aircraft agents are able to fly according a three dimensional flight plans or ATC vector instructions (i.e. heading, altitude and speeds orders). The aircraft aerodynamic model is based on a simplified point mass model that is described in [51]. In addition some navigation coordination tasks have been implemented on it (e.g. modifying arrival flight plan to perform a continuous descent and communicate this modification to the ATC agent via data message). In the same way, the ATC agent can detect missed separation conflict and provide primary conflicts resolution

Then, this architecture is intended as later extensions in order to add new algorithms (i.e. conflict detection and resolution algorithms, arrival sequence algorithms, etc.), air-air and air-ground negotiations protocols, human-machine interfaces and decision-making support systems, etc.

7. Conclusions

A summary description of a proposed conceptual model that represents TBO scenarios as a multi-agent system has been presented in this chapter. The aim of this design was to illustrate how current agent-oriented methodologies have been successfully applied to achieve highly structured representations of these scenarios and, therefore, they are a powerful design tool previous to a full implementation of future operational concepts.

A practical and formal methodological approach has been used to analyse and design the mentioned scenarios in a structured and consistent manner. By means of an iterative top-down modelling process the detailed agents architecture were designed based on capabilities, plans, events, and data structures.

The ATC view point was also described in this chapter through its inner architecture design. This architecture is oriented to execute several processes in order to plan, execute or modify trajectories in a coordinated way.

This approach has been based on guidelines provides by the recent Prometheus methodology. It has showed to be a suitable methodology for building models of next-generation ATM systems in the light of the following features:

- This approach achieves the goals of obtaining a highly structured model with several levels of abstractions. This structured nature allows a suitable integrity and consistence verification of the model on each of its three design phases: system specification, architecture design and detailed design.
- Also, Prometheus methodology provides proper guidelines for obtaining a system specification based on a goals hierarchical structure. These goals were identified from an organized set of scenarios that illustrate several aspects of the operational behaviour of the system. Goal at lowest level are used for defining required functionalities of the system as well as its main data, actions and perceptions.

- The overall system architecture combines information about roles of the air traffic entities with communication protocols that agents needs in order to improve their knowledge about the environment, agents states and intentions. Protocols are, also, a key aspect into the agent negotiation processes to achieve their respective goals.
- It illustrates how agent processes can be implemented by a set of several plans into agent. Plans are organized into several capabilities. Besides, this modular agent architecture based on plans allows a latter inclusion of new plans for implementing new procedures and functionalities. Moreover it is particularly important for obtaining robust software models.
- Finally, the model connects in a natural way those components that represent the dynamic perspective from those one than give a structural vision of the model. Protocols and processes, that model the dynamic behaviour, represent the core of the procedures. In the other hand capabilities are a high level representation of the systems required by ATC, aircraft and other air traffic entities to execute their task in a procedural manner.

After this first version of a simulation platform has been implemented and validated, new procedures, functionalities and underlying models will be included to be analysed as they are designed and included in the system following the described methodology. Furthermore, directions for future works include the extension of this conceptual model to gate-to-gate operations, as well as obtaining a full executable model for analytical simulation according to the described requirements.

Author details

José Miguel Canino
University of Las Palmas de Gran Canaria, Spain

Juan Besada Portas
University Polytechnic of Madrid, Spain

José Manuel Molina and Jesús García
University of Carlos III of Madrid, Spain

Acknowledgement

This work was funded by Spanish Ministry of Economy and Competitiveness under grant TEC2011-28626 C01-C02, and by the Government of Madrid under grant S2009/TIC-1485 (CONTEXTS).

8. References

[1] Ljungberg M, Lucas A (1992) The OASIS air-traffic management system. Proceedings of the Second Paciffic Rim International Conference on Artificial Intelligence, PRICAI.

[2] Garcia, J. L. (1990). MAESTRO-A metering and spacing tool. American Control Conference, 1990 (pp. 502–507).
[3] Schubert, M. (1990). COMPAS system concept. The COMPAS System in the ATC Environment 19 p(SEE N 92-19041 10-04).
[4] NASA (2012), Overview of CTAS. http://www.aviationsystemsdivision.arc.nasa.gov/research/foundations/sw_overview.shtml#overview
[5] van Gool M, Schoröter H (1999). PHARE Final Report, EUROCONTROL, Bruxelles. Availlable: http://www.eurocontrol.int/phare/gallery/content/public/documents/99-70-09pharefinal10.pdf
[6] RTCA (1995) Report of the RTCA Board of Directors' Select Committee on Free Flight, RTCA, Inc. Washington DC.
[7] Wilson, I. A. . (2007). 4-Dimensional Trajectories and Automation Connotations and Lessons learned from past research. Integrated Communications, Navigation and Surveillance Conference, 2007. ICNS'07 (pp. 1–10).
[8] Brooker P (2008) SESAR and NextGen: Investing in New Paradigms. Journal of Navigation, vol. 61, no. 2, pp. 195–208
[9] NASA (2006) Next Generation Air Transportation System (NGATS) Air Traffic Management (ATM)-Airspace Project. Reference Material. Available: http://cafefoundation.org/v2/pdf_tech/NASA.Aeronautics/PAV.NASA.ARMD.NGATS.pdf
[10] [10] NLR (2002) The NLR Air Traffic Control Research Simulator (NARSIM) Available: http://www.nlr.nl/documents/flyers/f075-07.pdf
[11] Weske R and Danek G (1993) Pseudo Aircraft Systems- A multi-aircraft simulation system for air traffic control research. AIAA Flight Simulation Technologies Conference, Monterey, CA, 1993, pp. 234–242. Available: http://www.aviationsystemsdivision.arc.nasa.gov/research/foundations/pas.shtml
[12] FAA (2009) Target Generation Facility. http://hf.tc.faa.gov/capabilities/tgf.htm
[13] Vic D (2011) ATC interactive for the future of Air Traffic Control. URL: http://users.skynet.be/atcsim/
[14] Prevot T (2002) Exploring the many perspectives of distributed air traffic management: The Multi Aircraft Control System MACS. Proceedings of the HCI-Aero, 149-154.
[15] Clari M, Ruigrok R, Heesbeen B and Groeneweg J (2002) Research Flight Simulation of Future Autonomous Aircraft Operations. Proceedings of Winter Simulation Conference, San Diego, USA.
[16] Sweet D N, Manikonda V, Aronson, J S, Roth K, Blake, M (2002) Fast-time Simulation System for Analysis of Advanced Air Transportation Concepts· Proceedings of the AIAA Modeling and Simulation Technologies Conference, Monterey, CA.
[17] Callantine T J, Homola J, Prevot T, Palmer E A (2006) Concept Investigation via Air-Ground Simulation with Embedded Agents. Modeling and Simulation Technologies Conference and Exhibit, Reston, VA, USA.

[18] Callantine TJ, Palmer EA, Homola J, Mercer J Prevot T (2006). Agent-Based Assessment of Trajectory-Oriented Operations with Limited Delegation. 25th Digital Avionics Systems Conf., Portland, OR.

[19] Wangermann J P, Stengel R F (1998) Principled negotiation between intelligent agents: a model for air traffic management. Artificial Intelligence in Engineering, 12(3), 177-187.

[20] Wangermann J P, Stengel R F (1999) Optimization and Coordination of Multiagent Systems Using Principled Negotiation. Journal of Guidance, Control and Dynamics, 22(1), 43-50.

[21] Wangermann, J P and Stengel R F (1996) Distributed optimization and principled negotiation for advancedair traffic management. Proceedings of the 1996 IEEE International Symposium on. pp. 156-161.

[22] Harper K A, Mulgund S S, Guarino S L, Mehta A V and Zacharias G L (1999) Air traffic controller agent model for Free Flight. AIAA Guidance, Navigation, and Control Conference and Exhibit, pp. 288-301, Portland, OR.

[23] Stengel R F (1993) Toward intelligent flight control. Systems, Man and Cybernetics. IEEE Transactions on, 23(6), 1699-1717.

[24] Belkin B L, Stengel, R F (1987) Cooperative rule-based systems for aircraft control. 26th IEEE Conference on Decision & Control, Los Angeles, CA

[25] Stengel R F, Niehaus A (1989). Intelligent guidance for headway and lane control. Engineering Applications of Artificial Intelligence, 2(4), 307-314.

[26] Shandy S, Valasek J (2001) Intelligent agent for aircracft collision avoidance. AIAA Guidance, Navigation, and Control Conference, Montreal, Canada.

[27] Rong J, 2002, Intelligent Executive Guidance Agent For Free Flight. AIAA-2002-15 . Reno, NV

[28] Rong J, Ding Y, Valasek J, Painter J (2003) Intelligent system design with fixed-base simulation validation for general aviation. Proc. IEEE International Symposium on Intelligent Control, Houston, Texas.

[29] Painter J H (2002) Cockpit multi-agent for distributed air traffic management. En AIAA Guidance, Navigation, and Control Conference and Exhibit. Monterey, CA.

[30] Ding Y, Rong J, Valasek, J (2003) Automation Capabilities Analysis Methodology for Non-Controlled Airports. AIAA Modeling and Simulation Technologies Conference and Exhibit. Austin, Texas.

[31] Satapathy G, Manikonda V (2004). Agent Infrastructures for Modeling and Simulation of CNS in the NAS. En Fairfax, VA..

[32] Canino J M , García J, Besasa J., Gómez L (2008) EATS: An Agent-Based Air Traffic Simulator. IADIS International Journal of Computer Science and Information Systems, Vol. 3, Issue 2.

[33] Iglesias CA, Garijo M ,Gonzalez J C, Velasci J R (1996) A Methodological Proposal for Multiagent Systems Development Extending CommonKADS http://citeseernj.nec.com/

[34] Giunchiglia F, Mylopoulos J, Perini A(2002) The Tropos Software Development Methodology: Processes, Models and Diagrams. 2002 Autonomous Agents and Multi-Agent Systems (AAMAS 2002), Bologna, Italy

[35] Nwana H S, Ndumu D T, Lee L C, Collis J C (1999) ZEUS: A Toolkit and Approach for Building Distributed Multi-Agent Systems. J. M. Bradshaw, ed., Proceedings of the Third International Conference on Autonomous Agents (Agents '99), ACM Press, Seattle, USA, pp. 360-361.
[36] Wood M F, DeLoach S A, (2001) An Overview of the Multiagent Systems Engineering Methodology. Agent-Oriented Software Engineering, Volume 1957 of LNCS, Berlin: Springer, January 2001, 207-221.
[37] Wooldridge M, Jennings N R, Kinny D (2000) The Gaia Methodology for Agent-Oriented Analysis and Design. Journal of Autonomous Agents and Multi-Agent Systems, 3, (3), 285-312
[38] Pavón J, Gómez-Sanz J (2006) INGENIAS web site: http:// grasia.fdi.ucm.es/ingenias/, consulted at December 2006.
[39] Cernuzzi L, Rossi G (2002) On the evaluation of agent oriented modeling methods. En Proceedings of Agent Oriented Methodology Workshop, Seattle, November.
[40] Sturm A, Shehor O (2004) A Framework for Evaluating Agent-Oriented Methodologies. Lecture notes in computer science, 94-109.
[41] Luck M M, Ashri R, D'Inverno M (2004) Agent-Based Software Development, Artech House.
[42] Padgham L Winikoff M (2004) Prometheus: A methodology for developing intelligent agents. Lecture Notes in Computer Science, 174-185.
[43] Padgham L, Thangarajah J, Winikoff M (2008) Prometheus Design Tool, (System Demonstration), Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence (AAAI), Chicago, Illinois, USA.
[44] Luigi F, Caire G, Greenwood D (2007) Developing Multi-Agent Systems with JADE. Wiley Series in Agent Technology, Hardcover.
[45] Foundation for Intelligent Physical Agents –FIPA- (2007) Standard Status Specifications, http://www.fipa.org/repository/standardspecs.html
[46] Sorensen, John A, (2000), Detailed Description for CE-11 Terminal Arrival: Self Spacing for Merging and In-trail Separation, NASA Ames Research Center and NASA Langley Research Center, Moffett Field, CA and Hampton, VA.
[47] Huget M P (2004) Agent uml notation for multiagent system design. IEEE Internet Computing, 2004, vol. 8, pp. 63–71.
[48] Winikoff M (2004) JACK TM intelligent agents: An industrial strength platform", Bordini et al. , pp. 175–193.
[49] Foundation for Intelligent Physical Agents. 24 Communicative Act Library Specification, Version J. http://www.24.org/specs/2400037/
[50] Bone RS (2005) Cockpit Display of Traffic Information (CDTI) Assisted Visual Separation (CAVS): Pilot Acceptability of a Spacing Task During a Visual Approach. 6 USA/Europe Air Traffic Management R&D Seminar. Baltimore, MD.
[51] Glover W., Lygeros J. (2004). A Multi-Aircraft Model for Conflict Detection and Resolution Algorithm Evaluation. Technical Report WP1, Deliverable D1.3. Distributed Control and Stochastic Analysis of Hybrid Systems Supporting Safety Critical Real-Time Systems Design Project (HYBRIDGE). European Commission, Brussels.